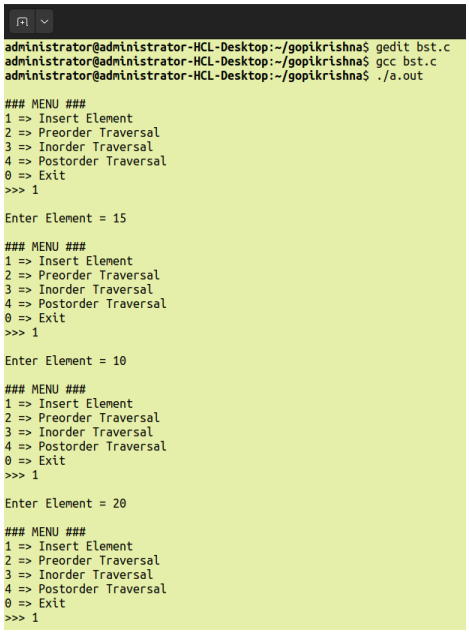
```
/*
        GOPIKRISHNA V
        S3 CSE A
        52
*/

#include <stdio.h>
#include <stdlib.h>

struct node
{
   int data;
   struct node *lchild;
   struct node *rchild;
};

void inorder(struct node *root)
{
   if (root == NULL)
      return;
   inorder(root -> lchild);
   printf("%d ",root -> data);
   inorder(root -> rchild);
}

void preorder(struct node *root)
{
   if (root == NULL)
      return;
   printf("%d ",root -> data);
   preorder(root -> lchild);
   preorder(root -> rchild);
}

void postorder(struct node *root)
{
   if (root == NULL)
      return;
   postorder(root -> lchild);
   postorder(root -> rchild);
   printf("%d ",root -> data);
}

struct node* new(int data)
{
   struct node* temp = (struct node*)malloc(sizeof(struct node));
   temp -> data = data;
   temp -> lchild = NULL;
```

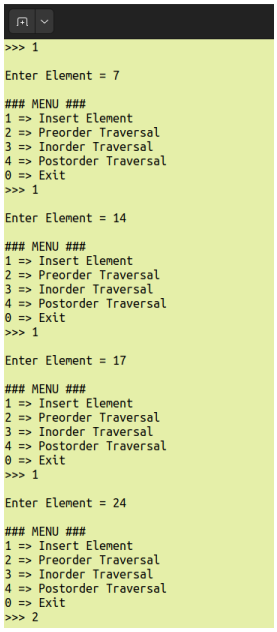
```
   temp -> rchild = NULL;
   return temp;
}

struct node* insert(struct node *node, int data)
{
   if (node == NULL){
      return new(data);
   }
   if (data < node -> data)
      node -> lchild = insert(node -> lchild, data);
   else if (data > node -> data)
      node -> rchild = insert(node -> rchild, data);
   return node;
}

void main()
{
   struct node* root = NULL;
   int run = 1,ans,data;

   while (run){
      printf("\n### MENU ###");
      printf("\n1 => Insert Element");
      printf("\n2 => Preorder Traversal");
      printf("\n3 => Inorder Traversal");
      printf("\n4 => Postorder Traversal");
      printf("\n0 => Exit");
      printf("\n>>> ");
      scanf("%d",&ans);
      printf("\n");

      switch (ans)
      {
      case 1:
         printf("Enter Element = ");
         scanf("%d",&data);
         root = insert(root, data);
         break;
      case 2:
         printf("PREORDER >>> ");
         preorder(root);
         printf("\n");
         break;
      case 3:
         printf("INORDER >>> ");
         inorder(root);
         printf("\n");
```

```
            break;
        case 4:
            printf("POSTORDER >>> ");
            postorder(root);
            printf("\n");
            break;
        case 0:
            printf("Exiting ...\n");
            run = 0;
            break;
        default:
            printf("Invalid\n");
            break;
        }
    }
}
```

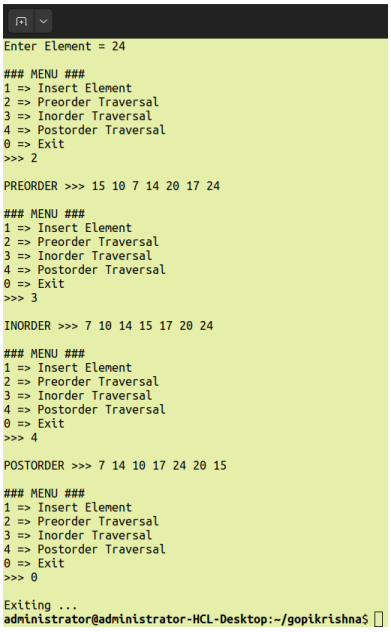
OUTPUT



```
administrator@administrator-HCL-Desktop:~/gopikrishna$ gedit bst.c
administrator@administrator-HCL-Desktop:~/gopikrishna$ gcc bst.c
administrator@administrator-HCL-Desktop:~/gopikrishna$ ./a.out

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1

Enter Element = 15

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1

Enter Element = 10

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1

Enter Element = 20

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1
```



```
>>> 1

Enter Element = 7

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1

Enter Element = 14

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1

Enter Element = 17

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 1

Enter Element = 24

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 2
```



```
Enter Element = 24

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 2

PREORDER >>> 15 10 7 14 20 17 24

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 3

INORDER >>> 7 10 14 15 17 20 24

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 4

POSTORDER >>> 7 14 10 17 24 20 15

### MENU ###
1 => Insert Element
2 => Preorder Traversal
3 => Inorder Traversal
4 => Postorder Traversal
0 => Exit
>>> 0

Exiting ...
administrator@administrator-HCL-Desktop:~/gopikrishna$
```